1 **Efficient Method for Multiplication Over Galois Fields**

2

3 ## Technical Field

4 The present invention relates to encryption algorithms and in particular to the

5 Advanced Encryption Standard (AES) issued by the National Institute of Standards and

6 Technology as part of the Federal Information Processing Standards (FIPS) Publication.

7 The AES specifies an FIPS approved cryptographic algorithm that can be used to

8 protect electronic data. More particularly, the present invention relates to an efficient

9 method of implementation of the AES encryption process.

10

11 ## Background of the Invention

12 Information shared between government agencies, and between government

13 agencies and contractors, is often sensitive. Such information is generally classified

14 according to guidelines established by the government agency involved. When such

15 classified information requires transmission between secured facilities (e.g., phone

16 conversations, FAXes, or transmission of computer files), a means must be exercised to

17 prevent the information from being intercepted. Also, communications between

18 individuals often include personal or business related content that that the individuals

19 intend to maintain as private.

20 The National Institute of Standards and Technology has established the

21 Advanced Encryption Standard (AES) as the approved cryptographic algorithm for such

22 transmission. Copies of the AES are available from the National Technical Information

23 Service (NTIS), 5285 Port Royal Road, Springfield, VA 22161.

24 The AES specifies the Rijndael algorithm, a symmetric block cipher that can

25 process data blocks of 128 bits, using cipher keys with lengths of 128, 192, and 256

26 bits. The input and output for the AES algorithm each consist of sequences of 128 bits

27 (digits with values of 0 or 1). The basic unit for processing in the AES algorithm is a

28 byte. The input, output and Cipher Key bit sequences are processed as arrays of bytes

29 that are formed by dividing these sequences into groups of eight contiguous bits to form

1 the arrays of bytes.

2   All byte values in the AES algorithm will be presented as the concatenation of its

3 individual bit values (0 or 1) between braces in the order $\{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$.

4 These bytes are interpreted as finite field elements using a polynomial representation:

5 $$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 = \sum_{i=0}^{7} b_ix^i \,.$$

6 For example, $\{01100011\}$ identifies the specific finite field element $x^6 + x^5 + x + 1$.

7   Internally, the AES algorithm's operations are performed on a two-dimensional

8 array of bytes called the State array. The State array consists of four rows of bytes,

9 each containing four bytes. At the start of the Cipher, or of the Inverse Cipher, the input

10 (a two dimensional array of bytes) is copied into the State array. The Cipher or Inverse

11 Cipher operations are then conducted on this State array, after which its final value is

12 copied to the output (a two dimensional array of bytes).

13   All bytes in the AES algorithm are interpreted as finite field elements. Finite field

14 elements can be added and multiplied, but these operations are different from those

15 used for normal numbers.

16   In the polynomial representation, multiplication (denoted by •) in a Galois field

17 (256), i.e., GF(2^8), corresponds with the multiplication of polynomials modulo m(x),

18 where m(x) is an irreducible polynomial of degree 8. A polynomial is irreducible if its

19 only divisors are one and itself. For the AES algorithm, this irreducible polynomial is:

20 $$m(x) = x^8 + x^4 + x^3 + x + 1 \,,$$

21 or 1{1b} in hexadecimal notation.

22   For example, $\{57\} \bullet \{83\} = \{c1\}$, because

23 $(x^6 + x^4 + x^2 + x + 1)\,(x^7 + x + 1) \quad = \quad x^{13} + x^{11} + x^9 + x^8 + x^7 +$

24 $\qquad\qquad\qquad\qquad\qquad\qquad\qquad x^7 + x^5 + x^3 + x^2 + x +$

25 $\qquad\qquad\qquad\qquad\qquad\qquad\qquad x^6 + x^4 + x^2 + x + 1$

26 $\qquad\qquad = \quad x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$

27

1 and

2 $$x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1 \text{ modulo } (x^8 + x^4 + x^3 + x + 1)$$

3 $$= x^7 + x^6 + 1.$$

4 The modular reduction by $m(x)$ ensures that the result will be a binary polynomial

5 of degree less than 8, and thus can be represented by a byte. However, there is no

6 simple operation at the byte level that corresponds to this multiplication.

7 The multiplication defined above is associative, and the element {01} is the

8 multiplicative identity. For any non-zero binary polynomial $b(x)$ of degree less than 8, the

9 multiplicative inverse of $b(x)$, denoted $b^{-1}(x)$, can be found as follows: the extended

10 Euclidean algorithm described in the CRC Press Handbook of Applied Cryptography,

11 published in 1997, on pages 81-83, is used to compute polynomials $a(x)$ and $c(x)$ such

12 that:

13 $$b(x)a(x) + m(x)c(x) = 1.$$

14 Hence, $a(x) \bullet b(x) \bmod m(x) = 1$, which means:

15 $$b^{-1}(x) = a(x) \bmod m(x).$$

16 Moreover, it holds that:

17 $$a(x) \bullet (b(x) + c(x)) = a(x) \bullet b(x) + a(x) \bullet c(x).$$

18 It follows that the set of 256 possible byte values, with XOR used as addition and

19 the multiplication defined as above, has the structure of the finite field GF(2^8).

20 The Cipher is comprised of four individual transformations: SubBytes, ShiftRows,

21 MixColumns, and AddRoundKey. The Mixcolumns transform includes multiplication

22 over a GF(2^8), which multiplication is described above. The Inverse Cipher similarly

23 includes four individual inverse transformations: InvShiftRows, InvSubBytes,

24 InvMixColumns, and AddRoundKey, wherein InvMixColumns also includes

25 multiplication over a GF(2^8). Each call to MixColumns and to InvMixColumns results in

26 sixteen such multiplications. Further, each Cipher or Inverse Cipher operation requires

27 9, 11, or 13 calls to MixColumns or InvMixColumns respectively, where the number of

28 calls depends on the key length.

3

1   As described above, multiplication over the GF(2^8) requires the multiplication of

2   two polynomials, followed by a modulo operation, which is Mips intensive. Alternatively,

3   the multiplication over the GF(2^8) may be performed by table lookup which is much less

4   Mips intensive. Generally such table would comprise a table size of 256 X 256 which

5   equals 65,536 elements. However, in the case of the AES algorithm, one term in the

6   multiplication is limited to 6 values, so the table size is reduced to 6 X 256 (i.e., 1536

7   elements.) However, even reducing the table size to 1536 elements places a burden on

8   memory space.

9   What is needed is a way to avoid the Mips intensive actual multiplication without

10   placing a burden on memory space.

11

12   **Summary of the Invention**

13   The present invention addresses the above and other needs by providing an

14   improved method for multiplying terms over a Galois field (GF). Each Cipher or Inverse

15   Cipher operation of the Advanced Encryption Standard (AES) encryption algorithm

16   requires 9, 11, or 13 calls to Mixcolumns (or to InvMixColumns), and each call to the

17   MixColumns (or to the InvMixColumns), transform results in sixteen multiplications over

18   a Galois field. Know methods of multiplication over a Galois field require Mips intensive

19   multiplication of polynomials followed by a modulo operation or a memory burdening

20   conventional table lookup. The present invention provides an efficient alternative to

21   known methods. The improved method takes advantage of the fact that in the Galois

22   field, any non zero element X can be represented by a power of a primitive element P.

23   The improved method utilizes a 2 by 256 table wherein one row is made up of the 256

24   elements of the Galois field, and the other row is made up of the log base P of the

25   corresponding element. The logs base P of the terms being multiplied are looked up

26   and summed, and the anti-log of the sum is looked up in the same table.

27   In accordance with one aspect of the invention, the multiplication of elements of a

28   Galois filed is performed through table look-up using a 2 by 256 table. By utilizing a log

29   table look-up, the size of the table is reduced from 6 by 256 to 2 by 256. The improved

1 method thereby reduces the memory required to perform multiplication over a Galois

2 field.

3   It is a feature of the present invention to avoid the Mips intensive operations

4 sometimes utilized for multiplication of elements of a Galois filed. A known method of

5 multiplication of elements of a Galois field requires the multiplication of two polynomials

6 (in the case of the AES algorithm, 8 order polynomials) followed by a modulo operation.

7   Such operations are Mips intensive. The use of the log table of the present invention

8 reduces the Mips required and thus speeds up the operation.

9   For a better understanding of the present invention, together with other and

10 further aspects thereof, reference is made to the following description, taken in

11 conjunction with the accompanying drawings, and its scope will be pointed out in the

12 appended clams.

13

Brief Description of the Drawings

15 Figure 1A shows a flow chart for the cipher process;

16 Figure 1B shows a flow chart for the inverse cipher process;

17 Figure 2 depicts a more detailed flow chart for the MixColumns transformation;

18 and

19 Figure 3 shows a block diagram of a typical communications system utilizing the

20 present invention.

21

Description of the Preferred Embodiments

23 The present invention is now described in detail with reference to the drawings.

24 In the drawings, each element with a reference number is similar to other elements with

25 the same reference number independent of any letter designation following the

26 reference number. In the text, a reference number with a specific letter designation

27 following the reference number refers to the specific element with the number and letter

28 designation and a reference number without a specific letter designation refers to all

29 elements with the same reference number independent of any letter designation

1   following the reference number in the drawings.

2   It should also be appreciated that many of the elements discussed in this

3   specification may be implemented in hardware circuit(s), a processor executing

4   software code, or a combination of a hardware circuit and a processor executing code.

5   As such, the term circuit as used throughout this specification is intended to encompass

6   a hardware circuit (whether discrete elements or an integrated circuit block), a

7   processor executing code, or a combination of a hardware circuit and a processor

8   executing code, or other combinations of the above known to those skilled in the art.

9   The improved encryption processing method of the present invention provides an

10   efficient alternative to both a Mips intensive polynomial multiplication and to a

11   conventional table lookup, used to multiply terms over a finite field. Known encryption

12   methods take advantage of the characteristics of finite field mathematics. For example,

13   a field which is mapped back onto itself by defined addition and multiplication operations

14   has the advantage of guaranteeing that the result of addition or multiplication will have a

15   known number of bit representation. Thus, by improving the efficiency of multiplication

16   of a finite field, the present invention improves the efficiency of the encryption methods.

17   The improved method takes advantage of the fact that in known finite fields, any

18   non zero element X can be represented by a power of a primitive element P. The

19   improved method utilizes a 2 by n table wherein a first row is made up of the n elements

20   of the finite field, and a second row is made up of the log base P of the corresponding

21   element in the first row. The logs base P of terms being multiplied are looked up and

22   summed, and the anti-log of the sum is looked up in the same table. In a preferred

23   embodiment, the method of the present invention is applied to the multiplications over a

24   Galois field (256) of the AES algorithm described in detail in the Advanced Encryption

25   Standard (AES) issued by the National Institute of Standards and Technology as part of

26   the Federal Information Processing Standards (FIPS) Publication. The Advanced

27   Encryption Standard is incorporated herein by reference.

28   A flow chart of the cipher 10 used by the AES algorithm for encryption is shown

29   in Figure 1A. The cipher 10 comprises four transformations: the SubBytes transform

1   12, the ShiftRows transform 14, the MixColumns transform 16, and the AddRoundKey

2   transform 18.   An input (original data) is copied into a State array. The original data

3   could be a voice signal, a signal from a FAX machine, computer files, or any other

4   signal that requires encryption. The AddRoundKey transform 18 is performed on the

5   State array. The four transforms 12, 14, 16, and 18 are executed nine, eleven, or

6   thirteen times, followed by a single execution of transforms 12, 14, and 18. Lastly, the

7   State array is copied into an output (encrypted data).

8   A flow chart of the inverse cipher 20 used by the AES algorithm for encryption is

9   shown in Figure 1B.   The inverse cipher 20 comprises four transformations: the

10   InvShiftRows transform 22, the InvSubBytes transform 24, the AddRoundKey transform

11   18, and the InvMixColumns transform 28. An input (encrypted data) is copied into a

12   State array. The AddRoundKey transform 18 is performed on the State array. The four

13   transforms 22, 24, 18, and 28 are executed nine, eleven, or thirteen times, followed by a

14   single execution of transforms 22, 24, and 18. The State array is lastly copied into an

15   output (original data).

16   The processing performed by the MixColumns transform 16 (and the

17   InvMixColumns transform 28) is shown in a flow chart in Figure 2. The MixColumns

18   transform 16 operates on each column of the State array independently, treating each

19   column as a four-term polynomial with coefficients over $GF(2^8)$, as described in the AES

20   algorithm, and recursively transforming each column by multiplying each column by a

21   fixed four term polynomial a(x) to obtain an updated column, where a(x) is:

22        $a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$

23   The multiplication by a(x) can be written in the form of a transformation matrix being

24   multiplied times a column of the State array as:

25

26
$$
\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < Nb
$$

1

2 where c is the column number and Nb is 4, with the result that the four bytes in a

3 column are transformed by:

4

5 $$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$

6 $$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$

7 $$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$

8 $$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c})$$

9

10   Based on the definitions provided in the AES algorithm description, each term of

11 the result of the transformation includes two multiplies (represented by the $\bullet$ operator)

12 over the $GF(2^8)$. The definition of multiplication over $GF(2^8)$ comprises the multiplication

13 of two polynomials followed my a modulo operation, which combination of operations is

14 Mips intensive.

15   The MixColumns transform 16 includes multiplication by only two constant

16 elements, {02} and {03}. Similarly, the InvMixColumns transform 28 includes

17 multiplication by only four constant terms, {0b}, {0d}, {09}, and {0e}. There are therefore

18 a total of 6 constant elements that are always one of the elements in the multiplication

19 over the $GF(2^8)$ in the AES algorithm. Thus, the multiplication of polynomials could be

20 replaced by a conventional table look-up, requiring a 6 by 256 element table, however

21 even a relaxed requirement for a 6 by 265 table places a burden upon memory.

22   The present invention replaces both the polynomial multiplication, and the

23 conventional table lookup, by a 2 by 256 primitive power and log table, and the steps of:

24 looking up the log of the terms being multiplied, summing the logs, and looking up the

25 anti-log of the sum. The method of the present method avoids both the Mips intensive

26 multiplication, and the burden on memory of the 6 by 256 table. The other steps of the

27 AES algorithm may be carried out as they would normally be.

28   A flow chart of a communications system utilizing the present invention is shown

29 in Figure 3. Provide input 32 provides original data 34 to a first communications device

1   36. A Digital Signal Processor (DSP) 38 circuit residing within the first communications

2   device 36 performs cipher processing on the original data 32 to produce an encrypted

3   signal 40. The encrypted signal 40 is transmitted from the first communications device

4   36 to a second communications device 42. A second DSP 44 circuit residing within the

5   second communications device 42 performs inverse cipher processing on the encrypted

6   signal to recover the original data 34. The original data 34 is then provided to receive

7   output 46.

8   Those skilled in the art will recognize that any signal may be digitized and

9   processed by the communications system of Figure 3. Signal may also flow from the

10   second communications device 42 to the first communications device 36, and multiple

11   communications devices may take part in the overall system. These various modes of

12   operation are intended to some within the scope of the present invention.

13   Those skilled in the art will further recognize that other encryption methods may

14   include multiplication over a Galois field, or other finite fields. The use of a primitive

15   power and log table to reduce Mips or memory required to perform such multiplication in

16   any encryption method is intended to fall within the scope of the present invention.

17   While the invention herein disclosed has been described by means of specific

18   embodiments and applications thereof, numerous modifications and variations could be

19   made thereto by those skilled in the art without departing from the scope of the

20   invention set forth in the claims

21